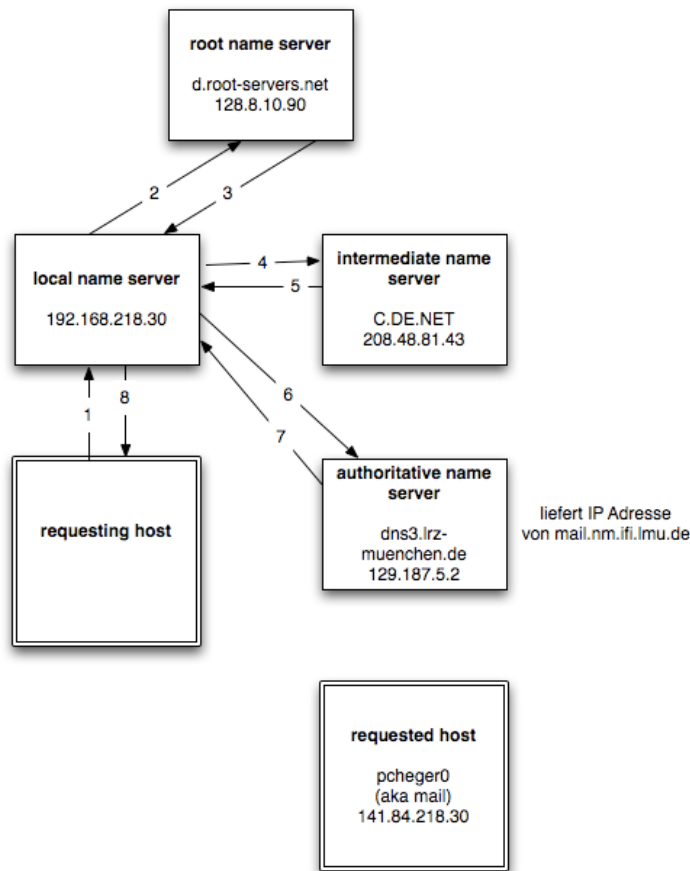


## Aufgabe 1)

### 1) Skizze



2) Die Anfrage ist iterativ. Dies kann man u.a. daran erkennen, dass der anfragende Host antworten von stets anderen Servern bekommt (Zeilen 16, 24, 29 und 39)

3)

a) Die DNS-Software (meistens BIND) muss mindestens die IP Adresse eines DNS Root-Server kennen. Diese werden in einer Datei des Betriebssystems gespeichert und regelmäßig aktualisiert.

b) Der Root-Server wird gefragt, ob er `mail.nm.ifl.lmu.de` kennt, was er aber nicht tut. Da der Root-Server seine Kinder kennt, kann es jedoch die DNS-Adressen der Server zurückgeben, die die Domäne `.de` verwalten.

4) `mail.nm.ifl.lmu.de` ist ein Alias für `pchege0.nm.ifl.lmu.de`, die unter der IP Adresse 141.84.218.30 zu finden ist.

5)

a) Das LRZ-München

b) `dns1.lrz-muenchen.de`.

`dns2.lrz-muenchen.de`.

`dns3.lrz-muenchen.de`.

c) Ja, da die 3 obigen eben authoritative Server sind (durch NS gekennzeichnet)

## Aufgabe 2)

Die HTML-Seite sei schon übertragen, es fehlen nur noch die Bilder.

Dazu muss der Client eine Verbindung aufstellen, eine Anfrage schicken, eine Antwort erhalten und die Verbindung wieder schließen. Mit den gegebenen Daten ergibt sich:

1)

- a) Client fragt nach SYN (10ms + 100ms), Server sagt SYN+ACK (10 + 100ms), Client macht ACK und schickt gleichzeitig Anfrage auf Bild 1 (10 + 100ms), Server gibt Bild zurück (10 + 100ms), Client sagt ACK + FIN (10 + 100ms), Server antwortet mit ACK + FIN (10 + 100ms) und Client wieder ACK (10 + 100 ms).  
Gesamt: 770ms, das ganze für 5 Bilder ergibt dann **3,85 Sekunden**
- b) Client fragt nach SYN (10ms + 100ms), Server sagt SYN+ACK (10 + 100ms), [Client macht ACK und schickt gleichzeitig Anfrage auf Bild 1 (10 + 100ms), Server gibt Bild zurück (10 + 100ms)]\*5, Client sagt ACK + FIN (10 + 100ms), Server antwortet mit ACK + FIN (10 + 100ms) und Client wieder ACK (10 + 100 ms).  
Gesamt: **1,65 Sekunden**
- c) TCP Aufbau noch ohne Pipelining:  
Client fragt nach SYN (10ms + 100ms), Server sagt SYN+ACK (10 + 100ms), Client macht ACK und schickt gleichzeitig Anfrage auf Bild 1 (10 ms), Bild 2 (10 ms), Bild 3 (10 ms), Bild 4 (10 ms), Bild 5 (10 ms), 60 ms später beginnt der Server mit dem Verschicken der Bilder, also wieder  $5 * 10 \text{ ms}$ , und 60ms nach dem Senden des letzten kommt das erste am Client an. Dieser verschickt 5 (10ms) ACK (das letzten mit FIN), 100 ms später Verschickt Server FIN+ACK die 110ms später ankommt, Client antwortet mit ACK (110 ms)  
Gesamt:  $220 + 110 + 110 + 150 + 110 + 110 = \mathbf{0,81 \text{ Sekunden}}$

2)

- a) Je größer die Netzverzögerung, desto mehr Leistungsverbesserung gibt es (ca. bis zu einem Faktor der die Anzahl der parallelen Verbindungen entspricht)
- b) Je größer die Objekte, desto größer die Leistungsverbesserung, da bei Pipelining die Sendezeit für die Objekte teilweise von der Netzverzögerung subtrahiert werden darf, da mehrere Objekte hintereinander versendet werden.
- c) Je mehr Objekte, desto größer die Verbesserung, da ja wie schon oben, teilweise einige Netzverzögerungen und Sendezeiten der Objekte ausfallen.

3)

- Slow Start, d.h. Fenstergröße wächst um eins für jedes ACK was zurückkommt:  
Client fragt nach SYN (10ms + 100ms), Server sagt SYN+ACK (10 + 100ms)  
(Fenstergröße = 2).  
Client schickt ACK + Anfrage für Bild 1 und Anfrage für Bild 2 (10 + 10 ms), 90 ms später bekommt Server erste Anfrage und schickt Bild 1 (10ms) und Bild 2 (10ms), die 90 ms später am Client antreffen. Wenn Bild1 ankommt, ist die Fenstergröße = 3 und Client schickt Anfrage auf Bild 3+ACK noch während Bild2 empfangen wird, danach schickt es eine Anfrage für Bild 4 (10ms). 90 ms später empfängt der Server die Anfrage für Bild 3 und schickt dies (10ms)(mit einer ACK) und 10 ms später auch Bild

4. 110 ms später ist Bild 4 empfangen, und der Client schickt ACK+FIN, was 110 ms später ankommt, Server schickt dann ACK + FIN (110 ms), Client schickt ACK (110)  
Gesamt: **1 Sekunde**

4)

- a) Bei Pipelining ist eine Art Queue nötig, die festhält für welche Elemente noch ACKs benötigt werden. Die Größe der Warteschlange wird von der Größe des Puffers bestimmt.
- b) Festlegung einer Idle-Zeit nach der die Verbindung falls unbenutzt wieder getrennt werden soll.
- c) keine zusätzlich Struktur nötig, da es ja keinen sichtbaren Zusammenhang zwischen den Verbindungen gibt